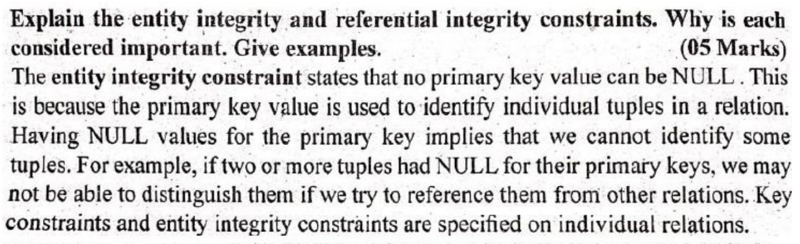
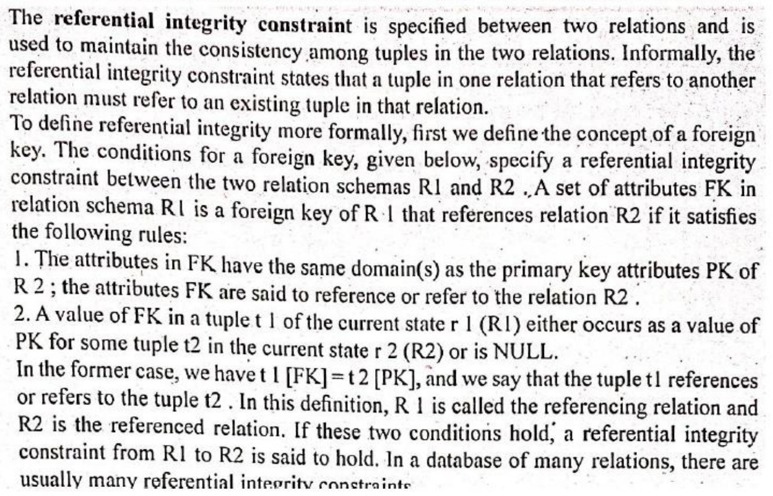
Characteristics of Relations **1.Ordering of Tuples in a Relation :**(1)A relation is defined as a set of tuples. Mathematically, elements of a set have no order among them; hence, tuples in a relation do not have any particular order. (2)Similarly, when tuples are represented on a storage device, they must be organized in some fashion, and it may be advantageous, from a performance standpoint, to organize them in a way that depends upon their content._**2.Ordering of Values within a Tuple :**(1)The order of attributes and their values is not that important as long as the correspondence between attributes and values is maintained. (2)A tuple can be considered as a set of (<attribute>,<value>) pairs, where each pair gives the value of the mapping from an attribute Ai to a value vi from dom(Ai). The ordering of attributes is not important, because the attribute name appears with its value. **3.Values and NULLs in the Tuples:**(1)Each value in a tuple is an atomic value; that is, it is not divisible into components. (2)An important concept is NULL values, which are used to represent the values of attributes that may be unknown or may not apply to a tuple. (3)NULL values has several meanings, such as value unknown, value exists but is not available, or attributedoes not apply to this tuple._**4.Interpretation (Meaning) of a Relation:**(1)Each tuple in the relation can then be interpreted as a fact or a particular instance of the assertion.(2)Each relation can be viewed as a predicate and each tuple in that relation can be viewed as an assertion for which that predicate is satisfied (i.e., has value true) for the combination of values in it.(3)Example:There exists a student having name Benjamin Bayer, having SSN 305-61-2435, having age 19, etc



Explain the entity integrity and referential integrity constraints. Why is each considered important. Give examples. **(05 Marks)**

The **entity integrity constraint** states that no primary key value can be NULL . This is because the primary key value is used to identify individual tuples in a relation. Having NULL values for the primary key implies that we cannot identify some tuples. For example, if two or more tuples had NULL for their primary keys, we may not be able to distinguish them if we try to reference them from other relations. Key constraints and entity integrity constraints are specified on individual relations.



The **referential integrity constraint** is specified between two relations and is used to maintain the consistency among tuples in the two relations. Informally, the referential integrity constraint states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation.

To define referential integrity more formally, first we define the concept of a foreign key. The conditions for a foreign key, given below, specify a referential integrity constraint between the two relation schemas R1 and R2 . A set of attributes FK in relation schema R1 is a foreign key of R 1 that references relation R2 if it satisfies the following rules:

1. The attributes in FK have the same domain(s) as the primary key attributes PK of R 2 ; the attributes FK are said to reference or refer to the relation R2 .
2. A value of FK in a tuple t 1 of the current state r 1 (R1) either occurs as a value of PK for some tuple t2 in the current state r 2 (R2) or is NULL.

In the former case, we have t 1 [FK] = t 2 [PK], and we say that the tuple t1 references or refers to the tuple t2 . In this definition, R 1 is called the referencing relation and R2 is the referenced relation. If these two conditions hold, a referential integrity constraint from R1 to R2 is said to hold. In a database of many relations, there are usually many referential integrity constraints

(06 marks)

Expalin Attribute Data Types for SQL?

Numeric data types include integer numbers of various sizes (INTEGER or INT, and SMALLINT) and floating-point (real) numbers of various precision (FLOAT or REAL, and DOUBLE PRECISION). Formatted numbers can be declared by using DECIMAL(*i,j*)—or DEC(*i,j*) or NUMERIC(*i,j*)—where *i*, the *precision*, is the total number of decimal digits and *j*, the *scale*, is the number of digits after the decimal point.

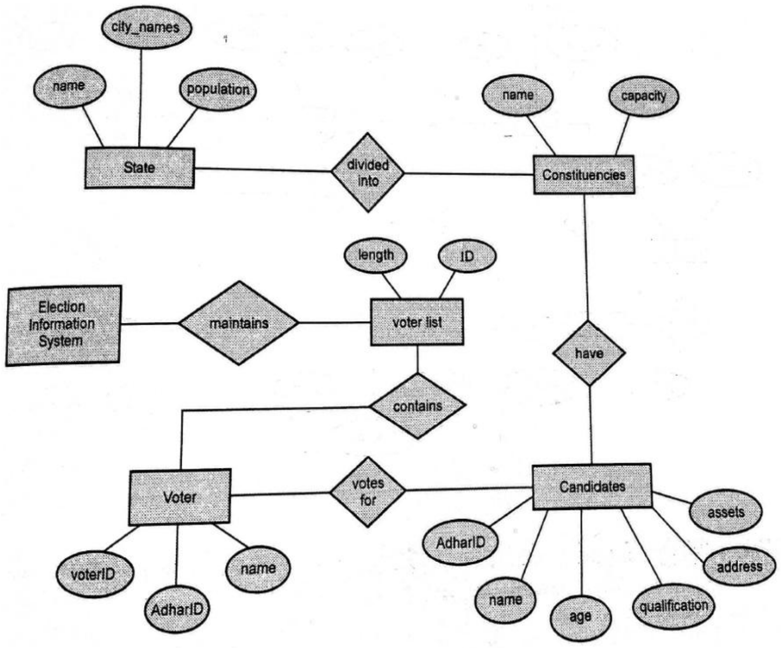
Character-string data types are either fixed length-CHAR (*n*) or CHARACTER (*n*), where *n* is the number of characters-or varying length-VARCHAR (*n*) or CHAR VARYING (*n*) or CHARACTER VARYING (*n*), where *n* is the maximum number of characters.

Bit-string data types are either of fixed length *n*—BIT(*n*)—or varying length—BIT VARYING(*n*), where *n* is the maximum number of bits.

Boolean data type has the traditional values of TRUE or FALSE. In SQL, because of the presence of NULL values, a three-valued logic is used, a third possible value for a Boolean data type is UNKNOWN.

The **DATE** data type has ten positions, and its components are YEAR, MONTH, and DAY in the form YYYY-MM-DD. The TIME data type has at least eight positions, with the components HOUR, MINUTE, and SECOND in the form HH:MM: SS.

A **timestamp** data type (TIMESTAMP) includes the DATE and TIME fields, plus a minimum of six positions for decimal fractions of seconds and an optional WITH TIME ZONE qualifier.


city_names
name
population
State
divided into
name
capacity
Constituencies
length
ID
Election Information System
maintains
voter list
have
contains
Voter
votes for
Candidates
assets
AdharID
address
voterID
name
AdharID
name
age
qualification